PSDesktop is a Java application that allows the user to drive EUROPA interactively by making use of the PSEngine client interface, and a number of prepackaged UI components.

Command Line arguments

PSDesktop takes 2 arguments :

- Debug Mode : must be "g" to load the debug version of the EUROPA libraries, or "o" to load the optimized version.
- bsh file (optional) : filename of the BeanShell file that is executed upon start

Running PSDesktop

There several ways to run PSDesktop (the last two have not been tested recently):

- run makeproject then go to the new project directory and run "ant"
- if you don't want to create a new project, copy .ant.psui.properties to your home directory, modify appropriately and run "ant" from $EUROPA_HOME/../../src/PLASMA/System/component/PSUI
- create your own Java project from scratch and include nddl.jar,PSEngine.jar and PSUI.jar (all found under $EUROPA_HOME/dist/europa/lib) in your classpath. Also make sure that you LD_LIBRARY_PATH includes the library directory. You'll want to run org.ops.psui.PSDesktop.main()

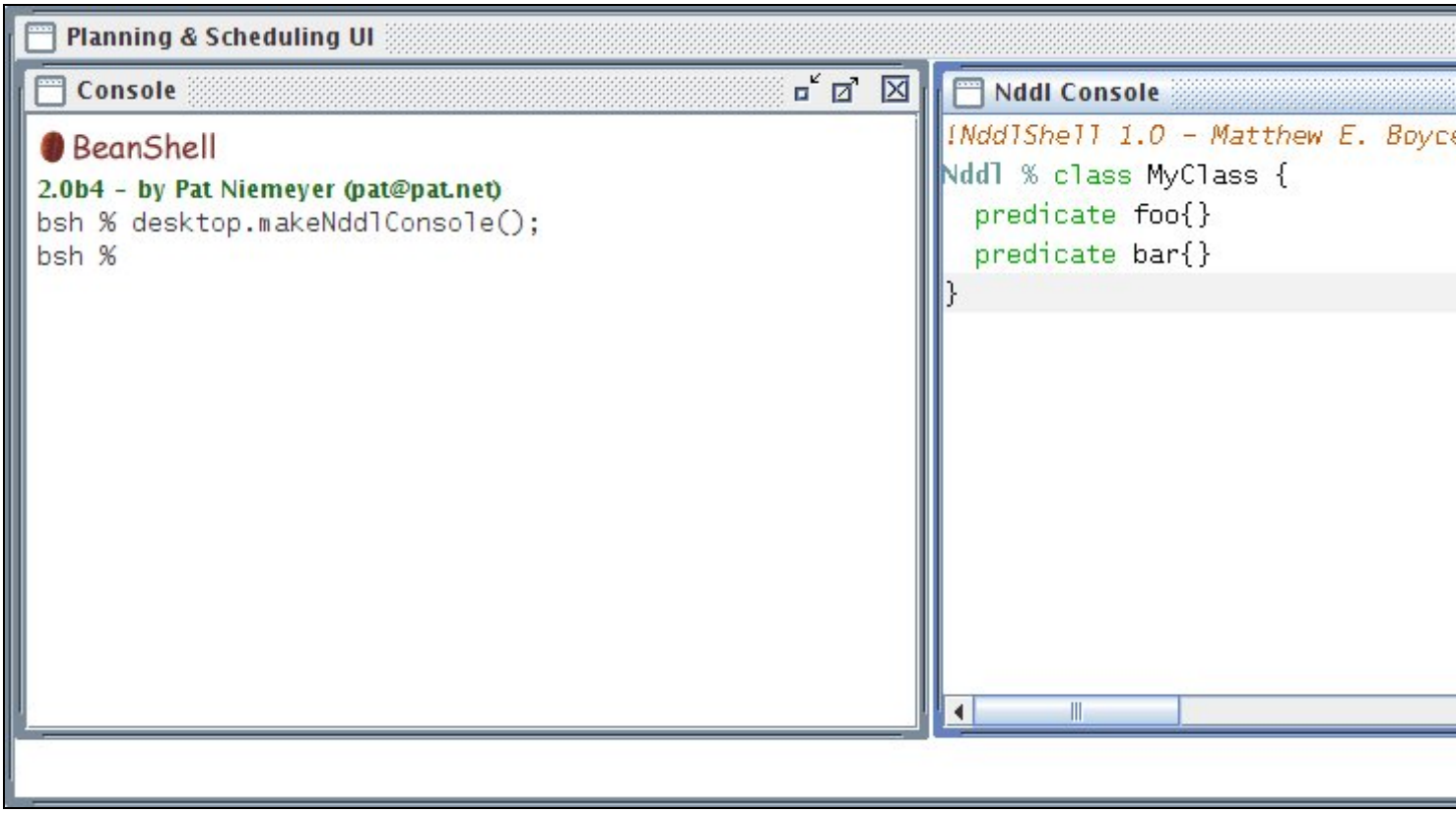
Once you run ant under any of these scenarios, you'll get the PSDesktop UI :

```
% cd $EUROPA_HOME/examples/Light
% ant
Buildfile: build.xml

init:

compile:

run:
     [echo] Running Light project
     [java]  autoWrite 0
     [java] [XMLInterpreter:InterpretedObject]Initialized variable:bulb1.mySwitch_ to OBJECT-LightSwitc
```



In the BeanShell console window you can type in Java statements that allow you to drive EUROPA interactively through its Java API (see below for the name of the exposed variables).

In the NDDL console you can type in NDDL statements that will be intrepreted as soon as you complete a valid NDDL statement and hit enter.

Variables exposed through BeanShell

In the BeanShell console, you'll have access to the following variables :

- PSEngine **psengine** : this will give you access to the EUROPA engine, you can create a solver, query the plan database, execute NDDL scripts, in general perform any task needed to drive EUROPA to load a model and create a plan. You can also use this interface to create your own custom solver, see for instance TSNQueensSolver.java
- PSDesktop **desktop** : this will give you access to many utility methods to create new desktop windows, display tables of tokens, create a solver, etc.

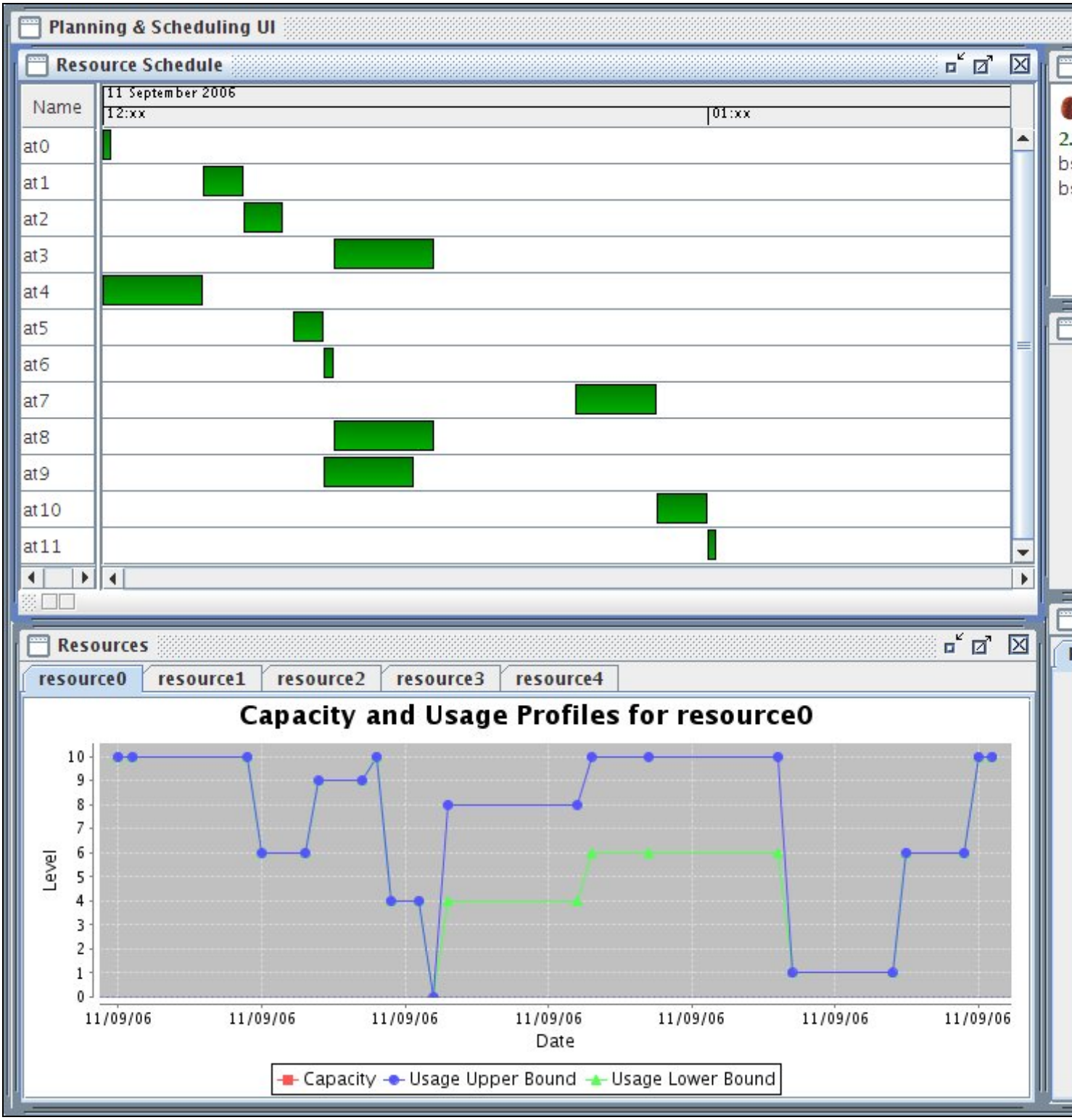
PSUI Components

The package here contains a number of components that make it easy to visualize your plan and interact with EUROPA : PLASMA/trunk/src/PLASMA/System/component

- PSGantt : allows you to see the tokens on a timeline as a gantt chart

- PSChart : allows you to see resource profiles as charts
- PSSolverDialog : allows you to drive a solver interactively and see its status as it tries to achieve the goals you specified for it
- ActionDetails and ActionViolation : allows you to easily display violation and detail information about actions in your plan as you mouse over actions in other components (for instance a gantt chart)

You can see how all these components can be used together in the bsh files for the EUROPA examples. For instance, below is a screenshot of the UI that is set up in UBO.bsh for a Resource Scheduling application. Using this component based approach it only took around 10 lines of code to set up all the components seen below :


Planning & Scheduling UI
Resource Schedule
Name
11 September 2006
12:xx
01:xx
at0
at1
at2
at3
at4
at5
at6
at7
at8
at9
at10
at11
Resources
resource0
resource1
resource2
resource3
resource4
Capacity and Usage Profiles for resource0
Level
0
1
2
3
4
5
6
7
8
9
10
11/09/06
11/09/06
11/09/06
11/09/06
11/09/06
11/09/06
11/09/06
Date
Capacity
Usage Upper Bound
Usage Lower Bound